Certificateless Key Agreement in the Standard Model *

Georg Lippold **Juan González Nieto**

Information Security Institute, Queensland University of Technology,
PO Box 2434, Brisbane, Queensland 4001, Australia
Email: {g.lippold,j.gonzaleznieto}@qut.edu.au

Abstract

We show how to construct a certificateless key agreement protocol from the certificateless key encapsulation mechanism introduced by Lippold et al. (2009*a*) in ICISC 2009 using the Boyd et al. (2008) protocol from ACISP 2008. We introduce the Canetti-Krawczyk (CK) model for certificateless cryptography, give security notions for Type I and Type II adversaries in the CK model, and highlight the differences to the existing e^{2}CK model discussed by Lippold et al. (2009*b*). The resulting CK model is more relaxed thus giving more power to the adversary than the original CK model.

Keywords: key agreement, key exchange, key encapsulation mechanism, certificateless, standard model, Diffie-Hellman

1 Introduction

Certificateless encryption introduced by Al-Riyami & Paterson (2003) is a variant of identity based encryption that limits the key escrow capabilities of the key generation centre (KGC), which are inherent in identity based encryption Boneh & Franklin (2003). Dent (2008) published a survey of more than twenty certificateless encryption schemes that focuses on the different security models and the efficiency of the respective schemes. In certificateless cryptography schemes, there are three secrets per party:

1. The key issued by the key generation centre (Dent (2008) calls it "partial private key"). We assume in the following that this key is ID-based, although it does not necessarily have to be ID-based.
2. The user generated private key x_{ID} (Dent calls it "secret value").
3. The ephemeral value chosen randomly for each protocol run.

Key agreement schemes provide an efficient means for two parties to communicate over an adversarial controlled channel. An overview of almost twenty identity based key agreement protocols has been compiled by Chen et al. (2007); they also provide security proofs for two of the surveyed protocols. Many ID-based schemes guarantee full privacy for both parties as long as the key generation centre (KGC) does not learn any of the ephemeral secrets used in computing the session key. But as Krawczyk (2005) points out, the leakage of ephemeral keys should not be neglected as they are usually precomputed and not stored in secure memory. In the context of identity based key agreement protocols, this means that as soon as the ephemeral key of either party leaks, a malicious KGC is able to compute the session key.

An overview of current certificateless key agreement schemes has been compiled by Swanson (2008). Certificateless key agreement schemes attempt to provide full privacy even if the ephemeral secrets of the parties leak to the key generation centre or if the key generation centre actively interferes with the messages that are exchanged (e.g. does a man-in-the-middle attack). The first certificateless key agreement scheme with a proof of security was recently published by Lippold et al. (2009*b*) in the random oracle model (ROM). Lippold et al. (2009*b*) describe why it is hard to construct and prove certificateless key agreement schemes. The scheme they propose is computationally very expensive and in the random oracle model. They leave the construction of an efficient protocol and the construction of a standard model secure protocol as open questions. In this paper, we would like to answer these two open problems. In ACISP 2008, Boyd et al. (2008) showed how to construct identity based (ID) and public key based (PK) authenticated key agreement (AKE) from key encapsulation mechanisms (KEM) secure in the respective model. In this paper, we extend the model to certificateless key encapsulation mechanisms (CL-KEM) and show how to construct an efficient CL-AKE secure in the standard model from the CL-KEM scheme in the standard model recently published by Lippold et al. (2009*a*).

The security model in this paper is a new development of the e^{2}CK security model defined by Lippold et al. (2009*b*) and the Canetti & Krawczyk (2001) security model. We extend the Canetti and Krawcyk (CK) model in Section 3 on page 3 to allow partial corruption of the long term secrets. We provide a meaningful merge between the e^{2}CK and the CK models for certificateless encryption that is strong enough to realistically model an adversary against the protocol and is at the same time flexible enough to allow a simple construction of a certificateless key agreement protocol in the standard model. The model is more relaxed due to the fact that we also allow partial corruption of parties. A certificateless key agreement protocol in this model is secure even if one party is fully corrupted and one party is partially corrupted. Additionally, we define the notion of weak Type I and weak Type II CK adversaries for certificateless key agreement. Type I adversaries model an outsider adversary that does not know the

*Research funded by the Australian Research Council through Discovery Project DP0666065

master secret key of the KGC; Type II adversaries model insider adversaries that possess the master secret key of the KGC. These definitions correspond to the weak Type I and weak Type II adversary definitions for certificateless encryption given by Dent (2008).

The main contributions of this paper are:

- First efficient provably secure protocol for certificateless key exchange in the standard model.
- New enhanced security model for certificateless key exchange based on the well-known Canetti-Krawczyk model, giving more power to the adversary.
- New definition of weak Type I and weak Type II certificateless adversaries in the Canetti-Krawczyk model.

2 Definitions

Def. 1 Min-entropy Gennaro et al. (2004)
Let χ be a probability distribution over A. The min-entropy of χ is the value

$$\textit{min-ent}(\chi) = \min_{x \in A: \Pr_\chi[x] \neq 0} (-\log_2(\Pr_\chi[x])) \quad (1)$$

If χ has min-entropy t, then for all $x \in A : \Pr_\chi[x] \leq 2^{-t}$.

Def. 2 Strong randomness extractor Nisan & Zuckerman (1996)
A family of efficiently computable hash functions $\mathcal{H} = \{h_\kappa : \{0,1\}^n \to \{0,1\}^k | \kappa \in \{0,1\}^d\}$ is called a strong (m, ϵ)-randomness extractor, if for any random variable X over $\{0,1\}^n$ that has min-entropy at least m, if κ is chosen uniformly at random from $\{0,1\}^d$, and R is chosen uniformly at random from $\{0,1\}^k$, the two distributions $\langle \kappa, h_\kappa(X)\rangle$ and $\langle \kappa, R\rangle$ have statistical distance ϵ, that is

$$\frac{1}{2} \sum_{x \in \{0,1\}^k} |\Pr[h_\kappa(X) = x] - \Pr[R = x]| = \epsilon \quad (2)$$

Def. 3 Pseudorandom Function Family(PRF)
Let $\mathcal{F} = \{f_s\}_{s \in S}$ be a family of functions for security parameter $k \in \mathbb{N}$ and with seed $s \in S = S(k)$. Let C be an adversary that is given oracle access to either f_s for $s \xleftarrow{\$} K$ or a truly random function with the same domain and range as the functions in $\mathcal{F}$. $\mathcal{F}$ is said to be pseudorandom *if C's advantage in distinguishing whether it has access to a random member of $\mathcal{F}$ or a truly random function is negligible in k, for all polynomial-time adversaries C. That is,*

$$\mathbf{Adv}^{p\text{-}rand}_{\mathcal{F},C}(k) = \Big| \Pr[C^{F_s(\cdot)}(1^k) = 1] - \Pr[C^{Rand(\cdot)}(k) = 1] \Big| \quad (3)$$

is negligible in k.

Def. 4 Decisional Diffie-Hellman (DDH)
Let F be a cyclic group of order p' generated by an element f. Consider the set $F^3 = F \times F \times F$ and the following two probability distributions over it:

$$\mathcal{R}_F = \{(f^a, f^b, f^c) : (a, b, c) \xleftarrow{\$} \mathbb{Z}_{p'}\} \quad (4)$$

and

$$\mathcal{DH}_F = \{(f^a, f^b, f^{ab}) : (a, b) \xleftarrow{\$} \mathbb{Z}_{p'}\} \quad (5)$$

We say the Decisional Diffie-Hellman Assumption *holds over $F = \langle f \rangle$ if the two distributions $\mathcal{R}_F$ and $\mathcal{DH}_F$ are indistinguishable by all polynomial-time adversaries $\mathcal{D}$. More precisely, for $k = |p'|$*

$$\mathbf{Adv}^{ddh}_{F,\mathcal{D}}(k) = \Big| \Pr[\mathcal{D}(1^k, \rho) = 1 | \rho \xleftarrow{\$} \mathcal{DH}_F] - \Pr[\mathcal{D}(1^k, \rho) = 1 | \rho \xleftarrow{\$} \mathcal{R}_F] \Big| \quad (6)$$

Def. 5 CL-KEM
A certificateless key encapsulation mechanism (CL-KEM) $\mathcal{E}$ = (Setup, KeyDer, UserKeyGen, Enc, Dec) consists of five polynomial-time algorithms:

Setup $(mpk, msk) \xleftarrow{\$} \mathsf{Setup}(1^k)$ *Given the security parameter $k \in \mathbb{N}$, the* Setup *algorithm returns a master public key mpk, and a master secret key msk.*

KeyDer $d_{\mathsf{ID}} \xleftarrow{\$} \mathsf{KeyDer}(mpk, msk, \mathsf{ID})$ *generates an ID-based private key d_{ID} from the master secret key msk for identity* ID.

UserKeyGen $(x_{\mathsf{ID}}, \beta_{\mathsf{ID}}) \xleftarrow{\$} \mathsf{UserKeyGen}(mpk, \mathsf{ID})$ *generates a user secret key x_{ID} and a user certificateless public key β_{ID} form the master public key and the identity.*

Enc $(C, K) \xleftarrow{\$} \mathsf{Enc}(mpk, \mathsf{ID}, \beta_{\mathsf{ID}})$ *generates the key K, and the certificateless key encapsulation C of K.*

Dec $K \xleftarrow{\$} \mathsf{Dec}(d_{\mathsf{ID}}, x_{\mathsf{ID}}, C)$ *extracts the key K from the certificateless encapsulation C using the ID-based private key d_{ID} and the user secret key x_{ID}.*

This definition of a CL-KEM is based on the definition given in Lippold et al. (2009a).

Def. 6 CL-KEM Security
In a CL-KEM environment, the adversary has access to the following oracles:

Reveal master key: *The adversary is given access to the master secret key.*

Reveal ID-based key(ID): *The adversary extracts the ID-based private key of party* ID.

Get user public key(ID): *The adversary obtains the certificateless public key for* ID. *If the certificateless key for the identity has not yet been generated, it is generated with the* user key gen *algorithm.*

Replace public key(ID, pk): *Party* ID*'s certificateless public key is replaced with* pk *chosen by the adversary. All communication (encryption, encapsulation) for Party* ID *will use the new public key.*

Reveal secret value(ID): *The adversary extracts the secret value x_{ID} that corresponds to the certificateless public key for party* ID. *If the adversary issued a* replace public key *query for* ID *before, $\perp$ is returned.*

Decapsulate(ID, C): *The adversary learns the decapsulation of C under* ID *or $\perp$ if C is invalid or if the adversary replaced the public key of* ID.

Decapsulate(ID, C, x): *The adversary learns the decapsulation of C under* ID *using the secret value x. The special symbol $\perp$ will be returned if C is invalid.*

Get challenge key encapsulation(ID^*): *The adversary requests a challenge key encapsulation and thus marks the transition from* **Oracles**$_1$ *to* **Oracles**$_2$ *in Experiment 7. The simulator returns a challenge key encapsulation as described in Experiment 7.*

The security of a CL-KEM scheme $\mathcal{E} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{UserKeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is defined by the following experiment:*

$$\begin{aligned}
&\textbf{Exp. Chall.}^{cl-kem-cca}_{\text{CL-KEM}\mathcal{M}}(k):\\
&(mpk, msk) \xleftarrow{\$} \textbf{CL-KEM Setup}(k)\\
&(\mathsf{ID}^*, state) \xleftarrow{\$} \mathcal{M}^{\textbf{Oracles}_1}(find, mpk)\\
&K_0^* \xleftarrow{\$} \mathcal{K}; (C^*, K_1^*) \xleftarrow{\$} \textbf{CL-KEM Enc}(pk, \mathsf{ID}^*)\\
&\gamma \xleftarrow{\$} \{0,1\}; K^* = K_\gamma^*\\
&\gamma' \xleftarrow{\$} \mathcal{M}^{\textbf{Oracles}_2}(\textit{guess}, K^*, C^*, state)\\
&\textit{Return } \gamma == \gamma'
\end{aligned} \tag{7}$$

The advantage an adversary $\mathcal{M}$ *has against a CL-KEM scheme is therefore expressed by*

$$\textbf{Adv}^{CL\text{-}KEM}_{\mathcal{M}}(k) = \left| \Pr\left[\textbf{Exp. Chall.}^{cl-kem-cca}_{\text{CL-KEM}\mathcal{M}}(k)\right] - 1/2 \right|$$

For a Type I *adversary* $\mathcal{M}$, **Oracles**$_1$ *and* **Oracles**$_2$ *mean access to all oracles listed above with the following limitations:*

1. *No* reveal master key *queries.*
2. C^* *must not be submitted to a* decapsulate *oracle under* ID^*.
3. *Not both (*reveal secret value *OR* replace public key*) AND* reveal ID-based key *oracles may be asked for* ID^*.

For a Type 2 *adversary* $\mathcal{M}$, **Oracles**$_1$ *and* **Oracles**$_2$ *are subject to the following limitations:*

1. **Oracles**$_1$ *and* **Oracles**$_2$ *now includes* reveal master key *as allowed query,*
2. C^* *must not be submitted to a* decapsulate *oracle under* ID^*.
3. reveal secret value *must never be asked for* ID^*,
4. **Oracles**$_1$ *must not include* replace public key *for* ID^*.

This definition stems from (Lippold et al. 2009a, Section 3.3)

Def. 7 Weak Perfect Forward Secrecy (wPFS) *A key-exchange protocol provides weak PFS (wPFS) if an attacker* $\mathcal{M}$ *cannot distinguish from random a key of any session for which the session and its matching session are clean*[1] *even if* $\mathcal{M}$ *has learned the private keys of both peers to the session (Krawczyk 2005, Definition 22)*

Def. 8 Key Compromise Impersonation (KCI) *We say that a KE-attacker* $\mathcal{M}$ *that has learned the private key of party* $\hat{A}$ *succeeds in a Key-compromise impersonation (KCI) attack against* $\hat{A}$ *if* $\mathcal{M}$ *is able to distinguish from random the session key of a complete session at* $\hat{A}$ *for which the session peer is uncorrupted and the session and its matching session (if it exists) are clean (Krawczyk 2005, Definition 20).*

[1] Roughly speaking *clean* is the same as *fresh* in Definition 9.

3 Formal definition of the security model

We want to use the Protocol by Boyd et al. (2008) to construct a certificateless key agreement in the standard model. Current existing security definitions for key agreement protocols were defined by Swanson (2008) in the e^{2}CK model and later improved by Lippold et al. (2009*b*). However, the protocol by Boyd et al. (2008) is proven in the Canetti-Krawczyk model. We continue to list the two security models and then discuss the major differences.

3.1 The e^{2}CK model

Lippold et al. (2009*b*) strengthened the e^{2}CK security model for certificateless key exchange that was introduced by Swanson (2008). We recapitulate the model here briefly.

Let $\mathcal{U} = \{U_1, \ldots U_n\}$ be a set of parties. The protocol may be run between any two of these parties. For each party there exists an identity based public key that can be derived from its identifier. There is a key generation centre that issues identity based private keys to the parties through a secure channel. Additionally, the parties generate their own secret values and corresponding certificateless public keys.

The adversary is in control of the network over which protocol messages are exchanged. $\Pi^t_{i,j}$ represents the t^{th} protocol session which runs at party i with intended partner party j. Additionally, the adversary is allowed to replace certificateless public keys that are used to compute the session key. The adversary does not have to disclose the private key matching the replaced certificateless public key to the respective party.

A session $\Pi^t_{i,j}$ enters an *accepted state* when it computes a session key $SK^t_{i,j}$. Note that a session may terminate without ever entering into an accepted state. The information of whether a session has terminated with acceptance or without acceptance is assumed to be public. The session $\Pi^t_{i,j}$ is assigned a partner ID $pid = (ID_i, ID_j)$. The session ID sid of $\Pi^t_{i,j}$ at party i is the transcript of the messages exchanged with party j during the session. Two sessions $\Pi^t_{i,j}$ and $\Pi^u_{j,i}$ are considered matching if they have the same pid (and sid).

The game runs in two phases. During the first phase of the game, the adversary $\mathcal{M}$ is allowed to issue the following queries in any order:

Send($\Pi^t_{i,j}, x$): If the session $\Pi^t_{i,j}$ does not exist, it will be created as initiator at party i if $x = \lambda$, or as a responder at party j otherwise. If the participating parties have not been initiated before, the respective private and public keys are created. Upon receiving the message x, the protocol is executed. After party i has sent and received the last set of messages specified by the protocol, it outputs a decision indicating accepting or rejecting the session. In the case of one-round protocols, party i behaves as follows:

- $x = \lambda$**:** Party i generates an ephemeral value and responds with an outgoing message only.
- $x \neq \lambda$**:** If party i is a responder, it generates an ephemeral value for the session and responds with an outgoing message m and a decision indicating acceptance or rejection of the session. If party i as an initiator, it responds with a decision indicating accepting or rejecting the session.

In this work, we require $i \neq j$, i.e. a party will not run a session with itself.

Reveal master key The adversary is given access to the master secret key.

Session key reveal($\Pi^t_{i,j}$): If the session has not accepted, it returns $\perp$, otherwise it reveals the accepted session key.

Reveal ID-based secret(i): Party i responds with its ID-based private key, e.g. $sH_1(ID_i)$.

Reveal secret value(i): Party i responds with its secret value x_i that corresponds to its certificateless public key. If i has been asked the *replace public key* query before, it responds with $\perp$.

Replace public key(i, pk): Party i's certificateless public key is replaced with pk chosen by the adversary. Party i will use the new public key for all communication and computation.

Reveal ephemeral key($\Pi^t_{i,j}$): Party i responds with the ephemeral secret used in session $\Pi^t_{i,j}$

We can group the key reveal queries into three types: the *reveal master key* and *reveal ID-based secret* queries try to undermine the security of the ID-based part of the scheme, the *reveal secret value* and *replace public key* queries try to undermine the security of the public key based part of the scheme, and the *reveal ephemeral key* query tries to undermine the security of one particular session.

We define the state *fully corrupt* as a session that was asked all three types of reveal queries: the *reveal master key* or *reveal ID-based secret*, the *reveal secret value* or the *replace public key*, and the *reveal ephemeral key* query.

Once the adversary $\mathcal{M}$ decides that the first phase is over, it starts the second phase by choosing a *fresh session* $\Pi^t_{i,j}$ and issuing a *Test($\Pi^t_{i,j}$)* query, where the *fresh session* and *test query* are defined as follows:

Def. 9 Fresh session
A session $\Pi^t_{i,j}$ is fresh if (1) $\Pi^t_{i,j}$ has accepted; (2) $\Pi^t_{i,j}$ is unopened (not being issued the session key reveal *query); (3) the session state at neither party participating in this session is* fully corrupted*; (4) there is no opened session $\Pi^u_{j,i}$ which has a matching conversation to $\Pi^t_{i,j}$.*

Test($\Pi^t_{i,j}$) The input session $\Pi^t_{i,j}$ must be fresh. A bit $b \in \{0,1\}$ is randomly chosen. If $b = 0$, the adversary is given the session key, otherwise it randomly samples a session key from the distribution of valid session keys and returns it to the adversary.

After the *test($\Pi^t_{i,j}$)* query has been issued, the adversary can continue querying except that the test session $\Pi^t_{i,j}$ should remain *fresh*. We emphasize here that partial corruption is allowed as this is a benefit of our security model. Additionally, *replace public key* queries may be issued to *any* party after the test session has been completed.

At the end of the game, the adversary outputs a guess $\hat{b}$ for b. If $\hat{b} = b$, we say that the adversary wins. The adversary's advantage in winning the game is defined as

$$Adv^{\mathcal{M}}(k) = \left|\Pr[\mathcal{M} \text{ wins}] - \frac{1}{2}\right|$$

3.2 The Canetti-Krawczyk model

We give a slightly shortened version of the CK model used by Boyd et al. (2008) and refer the reader to the paper for details. In the Canetti-Krawczyk (CK) model, a protocol π is modeled as a collection of n programs running at different parties $P_1, \ldots, P_n$. Each program is an interactive probabilistic polynomial-time (PPT) machine. A *session* is defined as an invocation of π at party P_i, and every party may have multiple sessions running concurrently. The communication network is controlled by the adversary $\mathcal{M}$, who is also a PPT machine. The adversary controls the message flow between the parties by activating a party P_i, which may be done in two ways:

1. An *establish session* (P_i, P_j, s) request where P_j is another party with whom the session is to be established, and s is the session ID string which uniquely identifies a session between the participants.
2. By means of an *incoming message* m with a specified sender P_j.

A *matching session* is defined by having two session (P_i, P_j, s) and (P'_i, P'_j, s') for that $P_i = P'_i$, $P_j = P'_j$ and $s = s'$. s is defined by the concatenation of the messages exchanged by the respective parties. $\mathcal{M}$ can ask any of the following queries:

corrupt(P_i) $\mathcal{M}$ learns the long term key of P_i.

session-key(P_i, P_j, s) $\mathcal{M}$ learns the session key of an accepted session for (P_i, P_j, s) at party P_i.

session-state(P_i, P_j, s) Returns the internal state information of party P_i with respect to session s with party P_j but does not include the long term key of P_j.

session-expiration(P_i, P_j, s) Erases the session key form for session s with P_j from the internal memory of P_i.

test-session(P_i, P_j, s) The challenger $\mathcal{B}$ selects a random bit b. If $b = 1$, then the correct session key is returned. Otherwise, a randomly chosen key from the probability distribution of the key space of the protocol is returned. This query may only be asked to a session that is not *exposed*, where an *exposed* session is defined as a session that has been asked either

- a *session-state* or *session-key* reveal query to this session or the matching session, or
- a corrupt query to either partner before the session expires at that partner.

3.3 Comparison of the two models

Although the *send* query in the e^{2}CK model roughly corresponds to the *activation* in the CK model, there are some subtle differences in the security models that we try to sort out in the following.

The CK model used by Boyd et al. (2008) does not consider partial corruption of the long term secret, as Boyd et al. only consider cases where a party has only one long term secret. On the contrary, in the e^{2}CK model used by Lippold et al. (2009*b*), each party has two long-term secrets which may be corrupted "individually".

Furthermore, Boyd et al. (2008) do not allow the adversary only to extract the randomness used in a specific run of the protocol, i.e. they do not allow *ephemeral key reveal* queries.

On the other hand, Boyd et al. allow *state reveal* queries in addition to the *session-key reveal* query that both models allow, but both queries are not allowed for the *test session*.

For the following, we propose the following meaningful mapping from one model to the other:

- Instead of using *ephemeral key reveal*, we allow *session state reveal* queries. We note that *session state reveal* queries must not target the test session, so they are of limited use to the adversary. Our goal in this paper is to extend the KEM-KEM-AKE construction by Boyd et al. (2008) to the certificateless case. As the proofs for KEM schemes do not allow the adversary to extract the randomness used during the key encapsulation (which would be the equivalent to an ephemeral key), ephemeral key reveal queries cannot be allowed in a protocol that uses a KEM as a building block for any other scheme. KEM schemes would be trivial to break if any adversary was allowed to recover the randomness used in the challenge query.
- However, we allow the adversary to corrupt the long term secrets of a party "individually". On the one hand this allows for Type II certificateless adversaries that correspond to a key generation centre for the ID-based scheme as described in Section 3.4, on the other hand we give more power to the adversary as the protocol must still be secure even if one party is fully corrupted and the other party is partially corrupted.

3.4 Adversaries against weakly CK-secure certificateless key agreement schemes

Both Swanson (2008) and Lippold et al. (2009*b*) give only definitions for adversaries in the e^{2}CK model. We give the first security definitions for an adversary in the Canetti-Krawczyk (CK) model against a weakly secure certificateless key agreement protocol. The Type I adversary models an outsider adversary that may corrupt parties but may not learn the master secret key of the key generation centre (KGC). The Type II adversary reflects a malicious but honest KGC.

Def. 10 Weak Type I CK-secure key agreement scheme
A certificateless key agreement scheme is Weak Type I CK-secure if every probabilistic, polynomial-time adversary $\mathcal{M}$ has negligible advantage in winning the game described in Section 3 on page 3 subject to the following constraints.

- *$\mathcal{M}$ may corrupt at most two long-term secrets of one party involved in the* test *session, and one long-term secret of the other party involved in the* test *session.*
- *$\mathcal{M}$ is allowed to replace public keys of any party; however, this counts as the corruption of one secret.*
- *$\mathcal{M}$ may not reveal the secret value of any identity for which it has replaced the certificateless public key.*
- *$\mathcal{M}$ is not allowed to ask* session key reveal *queries for session keys computed by identities where $\mathcal{M}$ replaced the identity's public key.*
- *$\mathcal{M}$ is allowed to replace public keys of any party after the* test *query has been issued.*
- *$\mathcal{M}$ is not allowed to ask* session state reveal *queries for sessions at identities where $\mathcal{M}$ replaced the identity's public key.*
- *$\mathcal{M}$ is not allowed to ask* session state reveal *queries for the* test *session or the matching session to the* test *session.*
- *$\mathcal{M}$ is not allowed to ask* ephemeral key reveal *queries.*

Def. 11 Weak Type II CK-secure key agreement scheme
A certificateless key agreement scheme is Weak Type II CK-secure if every probabilistic, polynomial-time adversary $\mathcal{M}$ has negligible advantage in winning the game described in Section 3 on page 3 subject to the following constraints.

- *$\mathcal{M}$ is given the master secret key s at the start of the game.*
- *$\mathcal{M}$ may corrupt at most one user secret key of the parties participating in the test session.*
- *$\mathcal{M}$ is allowed to replace the certificateless public key of any party; however, this counts as the corruption of a user secret key.*
- *$\mathcal{M}$ may not reveal the secret value of any identity for which it has replaced the certificateless public key.*
- *$\mathcal{M}$ is not allowed to ask session key reveal queries for session keys computed by identities where the identity's public key was replaced.*
- *$\mathcal{M}$ is allowed to replace public keys of any party after the test query has been issued.*
- *$\mathcal{M}$ is not allowed to ask* session state reveal *queries for sessions at identities where $\mathcal{M}$ replaced the identity's public key.*
- *$\mathcal{M}$ is not allowed to as* session state reveal *queries for the* test *session or the matching session to the* test *session.*
- *$\mathcal{M}$ is not allowed to ask* ephemeral key reveal *queries.*

4 The Boyd et al. Protocol with a CL-KEM

We recall the generic AKE protocol constructions from KEM schemes by Boyd et al. (2008) from ACISP'08. The first scheme does not offer weak perfect forward secrecy (wPFS) as described in Definition 7 on page 3 but offers key compromise impersonation (KCI) resistance as described in Definition 8 on page 3. It is shown in Table 1 on the following page. The second scheme adds an additional Diffie-Hellman to the first protocol and then achieves both weak perfect forward secrecy and KCI resistance. The protocol is shown in Table 2 on the next page. Boyd et al. (2008) prove these protocols secure even if the long term secret of one party is known to the adversary.

We use these protocols to construct a weakly CK-secure certificateless key agreement scheme in the standard model, by replacing the KEM scheme in the Boyd et al. (2008) construction with the certificateless KEM (CL-KEM) scheme recently proposed by Lippold et al. (2009*a*). Constructing a certificateless key agreement scheme in the standard model is an open problem considered by Lippold et al. (2009*b*).

A		B
$(C_A, K'_A) \xleftarrow{\$} \mathsf{enc}(pk, \mathsf{ID}_B)$		$(C_B, K'_B) \xleftarrow{\$} \mathsf{enc}(pk, \mathsf{ID}_A)$
	$\xrightarrow{A, C_A}$	
	$\xleftarrow{B, C_B}$	
$K'_B = \mathsf{dec}(pk, d_{\mathsf{ID}_A}, C_B)$		$(K'_A = \mathsf{dec}(pk, d_{\mathsf{ID}_B}, C_A))$
$K''_A = \mathsf{Exct}_\kappa(K'_A); K''_B = \mathsf{Exct}_\kappa(K'_B)$		$K''_B = \mathsf{Exct}_\kappa(K'_B); K''_A = \mathsf{Exct}_\kappa(K'_A)$
$s = A\|\|C_A\|\|B\|\|C_B$		$s = A\|\|C_A\|\|B\|\|C_B$
$K_A = \mathsf{Exct}_{K''_A}(s) \oplus \mathsf{Exct}_{K''_B}(s)$		$K_B = \mathsf{Exct}_{K''_B}(s) \oplus \mathsf{Exct}_{K''_A}(s)$
Erase all state except (K_A, s)		Erase all state except (K_B, S)

Table 1: Boyd et al. (2008) Protocol 1

A		B
$y_A \xleftarrow{\$} \mathbb{Z}_p; Y_A = f^{y_A}$		$y_B \xleftarrow{\$} \mathbb{Z}_p; Y_B = f^{y_B}$
$(C_A, K'_A) \xleftarrow{\$} \mathsf{enc}(pk, \mathsf{ID}_B)$		$(C_B, K'_B) \xleftarrow{\$} \mathsf{enc}(pk, \mathsf{ID}_A)$
	$\xrightarrow{A, C_A, Y_A}$	
	$\xleftarrow{B, C_B, Y_B}$	
$K'_B = \mathsf{dec}(pk, d_{\mathsf{ID}_A}, C_B)$		$(K'_A = \mathsf{dec}(pk, d_{\mathsf{ID}_B}, C_A))$
$K''_A = \mathsf{Exct}_\kappa(K'_A); K''_B = \mathsf{Exct}_\kappa(K'_B)$		$K''_B = \mathsf{Exct}_\kappa(K'_B); K''_A = \mathsf{Exct}_\kappa(K'_A)$
$K''_{AB} = \mathsf{Exct}_\kappa Y_B^{y_A}$		$K''_{BA} = \mathsf{Exct}_\kappa Y_A^{y_B}$
$s = A\|\|C_A\|\|Y_A\|\|B\|\|C_B\|\|Y_B$		$s = A\|\|C_A\|\|Y_A\|\|B\|\|C_B\|\|Y_B$
$K_A = \mathsf{Expd}_{K''_A}(s) \oplus \mathsf{Expd}_{K''_B}(s)$ $\oplus \mathsf{Expd}_{K''_{AB}}(s)$		$K_B = \mathsf{Expd}_{K''_B}(s) \oplus \mathsf{Expd}_{K''_A}(s)$ $\oplus \mathsf{Expd}_{K''_{BA}}(s)$
Erase all state except (K_A, s)		Erase all state except (K_B, S)

Table 2: Boyd et al. (2008) Protocol 2

In Table 1 and Table 2, the following notations are used:

- $\{\mathsf{Expd}_K(\cdot)\}_{K \in \mathbb{U}_1} : \{0,1\}^\sigma \to \mathbb{U}_2$ is a pseudorandom function family as in Definition 3 on page 2.
- $\mathsf{Exct}_\kappa(\cdot) : \mathbb{K} \to \mathbb{U}_1$ is a randomly chosen strong (m, ϵ)-randomness extractor as in Definition 2 on page 2 for appropriate m and ϵ.
- n_{orac} is the total number of sessions / oracles created by the adversary against the protocol.
- $\frac{1}{p}$ is the maximum probability that $C_1 = C_2$ where $(C_1, K_1) \xleftarrow{\$} \mathsf{enc}(pk, \mathsf{ID})$ and $(C_2, K_2) \xleftarrow{\$} \mathsf{enc}(pk, \mathsf{ID})$ for any identity ID.

5 Proving the derived protocol secure

Lippold et al. (2009*b*) point out that a "natural" combination of an ID-based key agreement protocol (ID-AKE) and a public key based key agreement protocol (PK-AKE) does not give full security in the CL-AKE setting. In the e^{2}CK model described by Lippold et al. (2009*b*) there are three secrets per party of which the adversary may corrupt any two to gain an advantage in breaking the CL-AKE protocol. The adversary may choose the secrets in its favour: by corrupting the ID-based secrets at party one an the PK secret at the other party, and the ephemeral secrets at both parties, both protocols are broken although each party still holds one uncompromised secret. We investigate if a CL-AKE from a CL-KEM plugged into the KEM-KEM construction by Boyd et al. (2008) provides reasonable security.

Obviously, as Protocol 1 from Boyd et al. (2008) does not achieve weak forward secrecy, the security of the protocol relies on the long-term keys of the respective parties. Thus, this type of protocol will not give resistance if all four long term secrets of both parties are compromised. We now turn to the security proof for the protocol in the reconsidered security model explained in Section 3.3 on page 4 and Section 3.4 on the previous page.

We follow the proof given in Boyd et al. (2008) closely and will only list the changes that are needed to prove the scheme secure against a certificateless adversary. Recall that the weak certificateless adversary may not ask session key or session state reveal queries for sessions at a party where the adversary replaced that party's certificateless public key nor may these queries be asked for the *test* session.

We give some intuition why the proof for the Boyd et al. (2008) protocol does still hold in the reconsidered security model from section 3.3 on page 4 and section 3.4 on the previous page. In a CL-KEM setting, there are only two secrets per party considered, as a KEM is a "receive only" protocol. The ephemeral secret used to construct the message is never disclosed to the adversary in a KEM protocol. The security of the KEM holds in the certificateless case if a party has at least one uncompromised secret, i.e. an uncompromised ID-based key or an uncompromised user secret key. Consider the CL-KEM-KEM-AKE setting, where the *test* session runs between party A with ID-based private key d_A and user secret key x_A and party B with ID-based private key d_B and user secret key x_B. There are now essentially three cases to distinguish:

1. A weak Type I CK-adversary $\mathcal{M}$ that corrupts both long-term secrets d_A and x_A of party A and one long-term secret of party B. In this case the security of the KEM at party B guarantees the security of the protocol.
2. A weak Type I CK-adversary $\mathcal{M}$ that corrupts only one long-term secret of party A but both long-term secrets d_B, x_B of party B. In this case the security of the KEM at party A guarantees the security of the protocol.

3. The case where the CK-adversary corrupts all long-term keys d_A, x_A, d_B, x_B of the respective parties. In this case Protocol 1 is broken, but Protocol 2 is still secure due to the additional Diffie-Hellman.

For Protocol 1 we propose the following theorem similar to Boyd et al. (2008):

Theorem 1 *Let $\mathcal{B}$ be any adversary against Protocol 1. Then the advantage of $\mathcal{B}$ against the SK-security (with partial WFS and KCI resistance) of Protocol 1 is:*

$$\mathbf{Adv}^{\mathrm{sk}}_{\mathcal{B}}(k) \leq \frac{n^2_{\mathrm{orac}}}{b} + 2n_{\mathrm{orac}}\left(\mathbf{Adv}^{\textit{CL-KEM-CCA}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\right)$$

The proof in Appendix A on the following page for Protocol 1 works for the cases 1 and 2 above and is very similar to the proof in Boyd et al. (2008). All games in the proof were left as in the original paper, only Game 3 was modified. In the certificateless setting, besides fully corrupting party A in case one, the adversary is also allowed to partially corrupt party B. In case two similarly the adversary is allowed to fully corrupt party B and partially corrupt party A. However, in both cases the CL-KEM scheme for the partially corrupted party (B in case 1 and A in case 2) is still secure, as by definition CL-KEM schemes tolerate partial corruption. Boyd et al. (2008) use any successful adversary against the KEM-KEM-AKE construction as an adversary against the KEM scheme in Game 3 of their proof. This proof technique carries through to the certificateless case in our modified security model as all oracle queries that the adversary may ask can still be answered by the challenger as described in Game 3 of Boyd et al. (2008).

For Protocol 2 on the previous page we have the following theorem in analogy to Boyd et al. (2008):

Theorem 2 *Let $\mathcal{B}$ be any adversary against Protocol 2. Then the advantage of $\mathcal{B}$ against the SK-security (with WFS and KCI resistance) of Protocol 2 is:*

$$\mathbf{Adv}^{\mathrm{sk}}_{\mathcal{B}}(k) \leq \max\left(2n^2_{\mathrm{orac}}\mathbf{Adv}^{\mathrm{ddh}}_{F,\mathcal{D}}(k) + 2\epsilon + 2\mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k), \frac{n^2_{\mathrm{orac}}}{p} + 2n_{\mathrm{orac}}\left(\mathbf{Adv}^{\textit{CL-KEM-CCA}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\right)\right).$$

As Boyd et al. (2008) do not alter Game 3 in their proof of Protocol 2 with respect to the proof of Protocol 1, the proof carries through o the CL-KEM case as well. We included their proof in Appendix B.

6 Conclusion

We give the first construction of an efficient certificateless key agreement scheme proven secure in the standard model. We use the existing KEM-KEM construction by Boyd et al. (2008) from ACISP 2008 to construct our scheme. We review existing security notions for certificateless key exchange and propose the new notion of Canetti-Krawczyk security for certificateless key agreement. We show that the KEM-KEM construction by Boyd et al. (2008) holds also for certificateless KEM schemes in the new certificateless Canetti-Krawczyk security model for certificateless key agreement.

References

Al-Riyami, S. S. & Paterson, K. G. (2003), Certificateless Public Key Cryptography, *in* C.-S. Laih, ed., 'ASIACRYPT', Vol. 2894 of *Lecture Notes in Computer Science*, Springer, pp. 452–473. Online available at http://eprint.iacr.org/2003/126.pdf. 1

Boneh, D. & Franklin, M. (2003), 'Identity based encryption from the Weil pairing', *SIAM Journal of Computing* **32**(3), 586–615. Online available at http://crypto.stanford.edu/~dabo/papers/bfibe.pdf. 1

Boyd, C., Cliff, Y., González Nieto, J. M. & Paterson, K. G. (2008), Efficient one-round key exchange in the standard model, *in* Y. Mu, W. Susilo & J. Seberry, eds, 'ACISP', Vol. 5107 of *Lecture Notes in Computer Science*, Springer, pp. 69–83. 1, 3, 4, 5, 6, 7

Canetti, R. & Krawczyk, H. (2001), Analysis of key-exchange protocols and their use for building secure channels, *in* B. Pfitzmann, ed., 'EUROCRYPT', Vol. 2045 of *Lecture Notes in Computer Science*, Springer, pp. 453–474. 1

Chen, L., Cheng, Z. & Smart, N. P. (2007), 'Identity-based key agreement protocols from pairings', *Int. J. Inf. Sec.* **6**(4), 213–241. 1

Dent, A. W. (2006), 'A note on game-hopping proofs', Cryptology ePrint Archive, Report 2006/260. http://eprint.iacr.org/2006/260. 8, 11

Dent, A. W. (2008), 'A survey of certificateless encryption schemes and security models', *International Journal of Information Security* **7**(5), 349–377. **URL:** *http://dx.doi.org/10.1007/s10207-008-0055-0* 1, 2

Gennaro, R., Krawczyk, H. & Rabin, T. (2004), Secure hashed diffie-hellman over non-ddh groups, *in* C. Cachin & J. Camenisch, eds, 'EUROCRYPT', Vol. 3027 of *Lecture Notes in Computer Science*, Springer, pp. 361–381. 2

Krawczyk, H. (2005), 'HMQV: A High-Performance Secure Diffie-Hellman Protocol', Cryptology ePrint Archive, Report 2005/176. http://eprint.iacr.org/2005/176. 1, 3

Lippold, G., Boyd, C. & González Nieto, J. M. (2009*a*), 'Efficient Certificateless KEM in the Standard Model', Cryptology ePrint Archive, Report 2009/451. http://eprint.iacr.org/2009/451, ICISC 2009 accepted paper. 1, 2, 3, 5

Lippold, G., Boyd, C. & González Nieto, J. M. (2009*b*), Strongly Secure Certificateless Key Agreement, *in* H. Shacham & B. Waters, eds, 'Pairing', Vol. 5671 of *Lecture Notes in Computer Science*, Springer, pp. 206–230. 1, 3, 4, 5, 6

Nisan, N. & Zuckerman, D. (1996), 'Randomness is linear in space', *J. Comput. Syst. Sci.* **52**(1), 43–52. 2

Swanson, C. M. (2008), 'Security in Key Agreement: Two-Party Certificateless Schemes', http://uwspace.uwaterloo.ca/bitstream/10012/4156/1/Swanson_Colleen.pdf. Master Thesis, University of Waterloo. Download 2009-01-29. 1, 3, 5

A Proof of Theorem 1

We now proceed to prove Theorem 1. The proof has two parts; the first part proves the security of Protocol 1 proves the security of Protocol 2 when the partner to the test session is not corrupted. The second part proves the security of Protocol 1 when the partner to the test session is corrupted (in this case, we require the test session to have a matching session by the time $\mathcal{B}$ finishes). Remember that we are only considering partial forward secrecy, and therefore $\mathcal{B}$ does not corrupt both the owner of the test session and the corresponding partner.

Throughout the proof, we call each session to be activated at a party an oracle. We denote the oracles with which $\mathcal{B}$ interacts Π_X^i where X is the name of a party and i is the number of the oracle. We number the oracles such that Π_X^i is the i^{th} oracle created by $\mathcal{B}$ out of all oracles created by $\mathcal{B}$ (i.e. if Π_X^i and Π_Y^j are two oracles, then $i = j$ implies $X = Y$). Also, for any party, X, the identity of that party is denoted e_X. We consider the following series of games with $\mathcal{B}$.

A.1 Case 1: Partner to the test-session is not corrupted

In this case, the partner to the test session is either not corrupted or only partially corrupted, and the owner of the test session may be fully corrupted, either prior to the session (as in a KCI attack), or after the session expires (as in a forward secrecy attack). This part of the proof uses the following series of games with $\mathcal{B}$.

Game 0. This game is the same as a real interaction with the protocol. A random bit b is chosen, and when $b = 0$, the real key is returned in answer to the test session query, otherwise a random key from $\mathbb{U}_2$ is returned.

Game 1. This game is the same as the previous one, except that if two different sessions output exactly the same message and have the same intended partner, the protocol halts.

Game 2. This game is the same as the previous one, except that before the adversary begins, a random value $m \xleftarrow{\$} \{1, 2, \ldots, n_{\text{orac}}\}$ is chosen. We call the m^{th} oracle to be activated the target oracle. If the target oracle is not the test oracle, the protocol halts and $\mathcal{B}$ fails and outputs a random bit. We denote the input message to the target oracle with C, the corresponding output message with C^*, the target oracle's owner with T and the target oracle's intended partner with T^*. Note that there may not be a matching test session activated at T^*.

Game 3. In this game, a random value K'^* is chosen. Whenever C^* is used as input to an oracle owned by T^*, the calculation of the key is modified so that K'^* is used in place of $\mathsf{dec}(pk, d_{T^*}, C^*)$; the message output by this oracle is calculated as usual. Similarly, K'^* is used instead of K'_T in the calculation of the session key by T.

The rest of the Game 3 is the same as Game 2. If $b = 1$, a random key from $\mathbb{U}_2$ is returned. Otherwise, K'^* is used in the computation of the test session as described above.

Game 4. This game is the same as the previous one, except that a random value $K''^* \xleftarrow{\$} \mathbb{U}_1$ is chosen and the use of $\mathsf{Exct}(K'^*)$ is replaced with K''^*.

Game 5. This game is the same as the previous one, except that whenever the value $\mathsf{Expd}_{K''^*}(s')$ for any s' would be used in generating keys, a random value from $\mathbb{U}_2$ is used instead (the same random value is used for the same value of s'; a different random value is chosen for different s').

A.2 Analysis of Games 0 to 2:

Let p_{sameMsg} be the probability of two or more sessions outputting the same message. We have

$$1 - p_{\text{sameMsg}} > 1 - \frac{n_{\text{orac}}^2}{b.}$$

Then

$$|\Pr[\sigma_0] - \Pr[\sigma_1]| < \frac{n_{\text{orac}}^2}{2b} \tag{8}$$

when $\frac{1}{2} > \frac{n_{\text{orac}}-1}{b} > 0$

This can be used to bound τ_0 as follows:

$$\begin{aligned} \tau_0 &= |2\Pr[\sigma_0] - 1| && (9)\\ &\leq 2\left(|\Pr[\sigma_0] - \Pr[\sigma_1]| + \left|\Pr[\sigma_1] - \frac{1}{2}\right|\right) && (10)\\ &\leq \frac{n_{\text{orac}}^2}{b} + \tau_1 && (11) \end{aligned}$$

In Game 2, the probability of the protocol halting due to an incorrect choice of m is $1 - \frac{1}{n_{\text{orac}}}$. Whether or not an abortion would occur in this game could be detected in the previous game if it also chose m in the same way. Therefore, we may use Dent's gamehopping technique as described in Dent (2006) to find:

$$\tau_2 = \frac{1}{n_{\text{orac}}}\tau_1 \quad \Rightarrow \quad n_{\text{orac}}\tau_2 = \tau_1 \tag{12}$$

and (11) gives

$$\tau_0 \leq \frac{n_{\text{orac}}^2}{b} + n_{\text{orac}}\tau_2 \tag{13}$$

A.3 Analysis of Game 3:

We now construct adversary $\mathcal{A}$ against the security of the CL-KEM, using $\mathcal{B}$. $\mathcal{A}$ is constructed such that when it receives the real key for the CL-KEM scheme, the view of $\mathcal{B}$ is the same as in Game 2, but if $\mathcal{A}$ receives a random CL-KEM key, the view of $\mathcal{B}$ is the same as in Game 3. Then, by the security of the CL-KEM scheme, we can claim these games are indistinguishable.

To begin, $\mathcal{A}$ is given the master public key pk. $\mathcal{A}$ passes this value as well as the description of $\mathsf{Exct}(\cdot)$, its key κ and $\{\mathsf{Expd}_K(\cdot)\}_{K\in\mathbb{U}_1}$ to $\mathcal{B}$. Recall that $\mathcal{A}$ has access to the corresponding oracles $\mathcal{O}_{\mathsf{KeyDer}}(\cdot)$ and $\mathcal{O}_{\mathsf{dec}}(\cdot,\cdot)$.

$\mathcal{A}$ runs as described in Game 2, except that when the target session is activated, $\mathcal{A}$ outputs e_{T^*} as the identity on which it wants to be tested. $\mathcal{A}$ receives a ciphertext C^* for T^* and key K'^*, which may be the decryption of C^* or may be a random CL-KEM key, each with equal probability. $\mathcal{A}$ then uses C^* as the output of the target session, modifies the calculation of keys so that K'^* is used in place of $\mathsf{dec}(pk, \mathsf{KeyDer}(pk, \alpha, e_{T^*}), C^*)$, and uses K'^* instead of K'_T to find the answer to the test session query when $b = 0$.

All legitimate queries made by $\mathcal{B}$ can still be answered by $\mathcal{A}$ using its oracles in as follows.

- A corrupt query on some identity e_X may be answered with $\mathcal{O}_{\mathsf{KeyDer}}(e_X)$.
- A partial corrupt query can be made by on the partner to the test session, T^*. This query can be answered with $\mathcal{O}_{\mathsf{KeyDer}}(T^*)$ for either the certificateless secret value or the ID-based private key but not both, as these queries are allowed in a certificateless KEM.
- $\mathcal{A}$ must maintain the session state of each oracle so that it may be returned in answer to session state reveal queries (session state reveal is not allowed on the test session or its matching session).
- Any message C_X to any party (including T^*) with identity e_X may be decrypted using $\mathcal{O}_{\mathsf{dec}}(e_X, C_X)$ to generate keys for reveal session key queries and the test query. Any party other than T^* can be decrypted with the private key that $\mathcal{A}$ generated for that party. A message M to T^* may be decrypted using $\mathcal{O}_{\mathsf{dec}}$.

When $\mathcal{B}$ halts and outputs its bit b', $\mathcal{A}$ halts and outputs $1-b'$. The probability that $\mathcal{A}$ is correct is $\Pr[\sigma_2]$ when K'^* is the real key for the CL-KEM message, and $1-\Pr[\sigma_3]$ when K'^* is not the key for the CL-KEM message. We can then find that:

$$\mathbf{Adv}^{\text{CL-KEM-CCA}}_{\mathcal{E},\mathcal{A}}(k) = \tag{14}$$

$$\left|2\left(\frac{1}{2}\left(\Pr[\sigma_2]+1-\Pr[\sigma_3]\right)\right)-1\right| \tag{15}$$

$$= \Pr[\sigma_2]-\Pr[\sigma_3] \tag{16}$$

$$\begin{aligned}
\tau_2 &= |2\Pr[\sigma_2]-1| && (17)\\
&\leq |2\Pr[\sigma_2]-2\Pr[\sigma_3]|+|2\Pr[\sigma_3]-1| && (18)\\
\tau_2 &\leq 2\mathbf{Adv}^{\text{CL-KEM-CCA}}_{\mathcal{E},\mathcal{A}}(k)+\tau_3 && (19)
\end{aligned}$$

A.4 Analysis of Game 4

We now consider an adversary, $\mathcal{D}$, against the security of the randomness extraction function. This adversary runs a copy of $\mathcal{B}$ and interacts with $\mathcal{B}$ in such a manner that it is the same as when $\mathcal{B}$ interacts with either Game 3 or 4. $\mathcal{D}$ receives a key κ for the randomness extraction function and a value R_1 such that either $R_1 = \mathsf{Exct}(X)$ for some $X \xleftarrow{\$} \mathcal{K}$ or $R_1 \xleftarrow{\$} \mathbb{U}_1$. $\mathcal{D}$ sets κ to be the public parameter used to key the randomness extraction function, and chooses the other public parameters according to the protocol. $\mathcal{D}$ runs as described for Game 4, except that $\mathcal{D}$ uses R_1 in place of K''^*. When $\mathcal{B}$ outputs it guess of the bit b, $\mathcal{D}$ outputs that $R_1 = \mathsf{Exct}(X)$ for some X if $\mathcal{B}$ is correct, and $\mathcal{D}$ outputs that $R_1 \xleftarrow{\$} \mathbb{U}_1$ otherwise. The probability that $\mathcal{D}$ is correct is $\frac{1}{2}(\Pr[\sigma_3]+1-\Pr[\sigma_4])$. By the security of the randomness extraction function, we have:

$$\begin{aligned}
\epsilon &\geq |2\Pr[\mathcal{D}\text{ correct}]-1| && (20)\\
&= |\Pr[\sigma_3]-\Pr[\sigma_4]| && (21)\\
\tau_3 &= |2\Pr[\sigma_3]-1| && (22)\\
&\leq |2\Pr[\sigma_3]-2\Pr[\sigma_4]|+|2\Pr[\sigma_4]-1| && (23)\\
&\leq 2\epsilon+\tau_4 && (24)
\end{aligned}$$

A.5 Analysis of Game 5

Now, we consider another adversary, $\mathcal{D}$', this time against the randomness expansion (or pseudorandom) function family $\{\mathsf{Expd}_K(\cdot)\}_{K\in\mathbb{U}_1}$. We define $\mathcal{D}$' to run a copy of $\mathcal{B}$, and to interact with $\mathcal{B}$ in such a manner that it is the same as when $\mathcal{B}$ interacts with with either Game 4 or 5. $\mathcal{D}$' receives the definition of the function family $\{\mathsf{Expd}_K(\cdot)\}_{K\in\mathbb{U}_1}$, and an oracle $\mathcal{O}(\cdot)$ which is either $\mathsf{Expd}_K(\cdot)$ for some value of K unknown to $\mathcal{D}$' or a truly random function. $\mathcal{D}$' runs a copy of the protocol for $\mathcal{B}$ in the same way as described for Game 4, except that whenever the value $\mathsf{Expd}_{K''^*}(s')$ for any s' would be used in generating keys, $\mathcal{D}$' uses the value $\mathcal{O}(s')$ instead. When $\mathcal{B}$ outputs it guess of the bit b, $\mathcal{D}$' outputs that its oracle is a member of the given function family if $\mathcal{B}$ is correct, and $\mathcal{D}$' outputs that its oracle is a truly random function otherwise. The probability that $\mathcal{D}$' is correct is $\frac{1}{2}(\Pr[\sigma_4]+1-\Pr[\sigma_5])$. By the security of the randomness expansion function we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{p-rand}}_{\mathcal{F},\mathcal{C}}(k) &\geq |2\Pr[\mathcal{D}'\text{ correct}]-1| && (25)\\
&= |\Pr[\sigma_4]-\Pr[\sigma_5]| && (26)
\end{aligned}$$

$$\begin{aligned}
\tau_4 &= |2\Pr[\sigma_4]-1| && (27)\\
&\leq |2\Pr[\sigma_4]-2\Pr[\sigma_5]|+|2\Pr[\sigma_5]-1| && (28)\\
&\leq 2\mathbf{Adv}^{\text{p-rand}}_{\mathcal{F},\mathcal{C}}(k)+\tau_5 && (29)
\end{aligned}$$

In Game 5, let us denote the key returned in the test session query with $R_1 \oplus \mathsf{Expd}_{K''_{T^*}}(s')$ when $b=0$, and R_2 when $b=1$, where R_1 and R_2 are chosen uniformly at random from $\mathbb{U}_2$. Now, R_2 is chosen independently of all other values in the protocol, so $\mathcal{B}$ can gain no information about R_2 directly; $\mathcal{B}$ can only gain information about R_2 by determining whether $b=0$ or $b=1$. Furthermore, when $b=0$, unless $\mathcal{B}$ can gain some information about R_1, the response to the test session query also looks random and is therefore indistinguishable from the case when $b=1$.

To gain information about R_1 from a source other than the test session query response, $\mathcal{B}$ must obtain the key of a session that has also used R_1 in the generation of its key. Now, if R_1 is used in the generation of a session's key, then that session must have had the same session identifier, and hence exchanged the same messages as the test session. Therefore, the session is either owned by T^* with intended partner T and received C^* as part of its input or is owned by T with intended partner T^* and had C^* as part of its output. However, such a session owned by T^* will match the test session and so not be subject to reveal key queries. Hence, $\mathcal{B}$ can gain no information about R_1, and so $\mathcal{B}$ can gain no information about b in Game 5, and therefore

$$\tau_5 = 0$$

Combining the results in equations (13), (19), (24) and (29) we conclude:

$$\tau_0 \leq \frac{n^2_{\text{orac}}}{b} + 2n_{\text{orac}}\Big(\mathbf{Adv}^{\text{CL-KEM-CCA}}_{\mathcal{E},\mathcal{A}}(k)+\epsilon + \mathbf{Adv}^{\text{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\Big) \tag{30}$$

A.6 Case 2: Partner to the test session is corrupted

Recall that we only consider partial weak forward secrecy for Protocol 1. Consequently we require that

1. the owner of the test session may only be partially corrupted by the adversary,
2. the adversary be passive in the protocol corresponding to the test session; that is there exists

a matching session to the test session at the intended partner by the time that the test session query is issued by the adversary; and

3. the partner to the test session be fully corrupted only after the matching session expires.

For this part of the proof we set up the following series of 6 games with $\mathcal{B}$. Game 0 and 1 are the same as in Case 1. Game 2 and 3 are analogous to Game 2 and 3 in Case 1 except that now our target oracle is the partner to the test session, and it is its input to the session key that is substituted by a random value. Game 4 and 5, which are used to prove the security of the session key derivation mechanism via randomness extraction and expansion also remain essentially the same.

Game 0. This game is the same as a real interaction with the protocol. A random bit b is chosen, and when $b = 0$, the real key is returned in answer to the test session query, otherwise a random key from $\mathbb{U}_2$ is returned.

Game 1. This game is the same as the previous one, except that if two different sessions output exactly the same message and have the same intended partner, the protocol halts.

Game 2. This game is the same as the previous one, except that before the adversary begins, a random value $m \xleftarrow{\$} \{1, 2, \ldots, n_{\text{orac}}\}$ is chosen. We call the m^{th} oracle to be activated the target oracle. If the target oracle is not the *partner* to the test oracle, the protocol halts and $\mathcal{B}$ fails and outputs a random bit. We denote the input message to the target oracle with C^*, the corresponding output message with C, the target oracle's owner with T^* and the target oracle's intended partner with T.

Game 3. In this game, a random value $K' \xleftarrow{\$} \mathcal{K}$ is chosen. Further a bit $c \in_R \{0, 1\}$ is chosen as a guess as to whether $\mathcal{B}$ fully corrupts T or T^*. Whenever C is used as input to an oracle owned by T, the calculation of the key is modified so that K' is used in place of $\mathsf{dec}(pk, d_T, C)$; the message output by this oracle is calculated as usual. Similarly, K' is used instead of K'_T in the calculation of the session key by T^*.

The rest of the Game 3 is the same as Game 2. If $b = 1$, a random key from $\mathbb{U}_2$ is returned. Otherwise, K' is used in the computation of the test session as described above.

Game 4. This game is the same as the previous one, except that a random value $K'' \xleftarrow{\$} \mathbb{U}_1$ is chosen and the use of $\mathsf{Exct}(K')$ is replaced with K''.

Game 5. This game is the same as the previous one, except that whenever the value $\mathsf{Expd}_{K''}(s')$ for any s' would be used in generating keys, a random value from $\mathbb{U}_2$ is used instead (the same random value is used for the same value of s'; a different random value is chosen for different s').

In the analysis that follows, we denote with σ'_i the event that the adversary is successful in Game i and with τ'_i the corresponding advantage.

A.7 Analysis of Games 0 to 2:

This analysis is the same as the that of Games 0 to 2 in Case 1. Thus,

$$\tau'_0 \leq \frac{n^2_{\text{orac}}}{b} + n_{\text{orac}}\tau'_2 \tag{31}$$

A.8 Analysis of Game 3:

This analysis is very similar to that of Game 3 in Case 1. We construct adversary $\mathcal{A}$ against the security of the CL-KEM, using $\mathcal{B}$. $\mathcal{A}$ is given the master public key pk and passes this value as well as the description of $\mathsf{Exct}(\cdot)$, its key κ and $\{\mathsf{Expd}_K(\cdot)\}_{K\in\mathbb{U}_1}$ to $\mathcal{B}$. $\mathcal{A}$ runs as described in Game 2, except that when the target oracle is activated, $\mathcal{A}$ outputs e_T as the identity on which it wants to be tested. $\mathcal{A}$ receives a ciphertext C for T and key K', which may be the decryption of C or may be a random CL-KEM key, each with equal probability. $\mathcal{A}$ then uses C as the output of the target oracle (and hence input to the test session). $\mathcal{A}$ modifies the calculation of keys so that K' is used in place of $\mathsf{dec}(pk, \mathsf{KeyDer}(pk, \alpha, e_T), C)$, and uses K' instead of K'_T to find the answer to the test session query when $b = 0$.

All legitimate queries made by $\mathcal{B}$ can still be answered by $\mathcal{A}$ using its oracles as follows.

- A corrupt query on some identity e_X may be answered with $\mathcal{O}_{\mathsf{KeyDer}}(e_X)$.
- A corrupt query targeting either the certificateless secret value or the identity based private key (but not both) for the owner of the test session can also be answered with $\mathcal{O}_{\mathsf{KeyDer}}(T)$. (Recall that the owner of the test session T must not be fully corrupted).
- $\mathcal{A}$ must maintain the session state of each oracle so that it may be returned in answer to session state reveal queries (session state reveal is not allowed on the test session or its matching session).
- Any message C_X to any party (including T) with identity e_X may be decrypted using $\mathcal{O}_{\mathsf{dec}}(e_X, C_X)$ to generate keys for reveal session key queries and the test query.

When $\mathcal{B}$ halts and outputs its bit b', $\mathcal{A}$ halts and outputs $1 - b'$. As in Case 1, the probability that $\mathcal{A}$ is correct is $\Pr[\sigma'_2]$ when K' is the real key for the CL-KEM message, and $1 - \Pr[\sigma'_3]$ when K' is not the key for the CL-KEM message. Hence,

$$\tau'_2 \leq 2\mathbf{Adv}^{\text{CL-KEM-CCA}}_{\mathcal{E},\mathcal{A}}(k) + \tau'_3 \tag{32}$$

A.9 Analysis of Game 4

This is the same as in Case 1. Thus,

$$\tau'_3 \leq 2\epsilon + \tau'_4 \tag{33}$$

A.10 Analysis of Game 5

This is the same as in Case 1. Thus,

$$\tau'_4 \leq 2\mathbf{Adv}^{\text{p-rand}}_{\mathcal{F},\mathcal{C}}(k) + \tau'_5 \tag{34}$$

A.11 Combining Results

Again using the same reasoning as in Case 1, we conclude that $\tau'_5 = 0$, and therefore combining equations (31), (32), (33) and (34) we have:

$$\tau'_0 \leq \frac{n^2_{\text{orac}}}{b} + 2n_{\text{orac}}\Big(\mathbf{Adv}^{\text{CL-KEM-CCA}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \mathbf{Adv}^{\text{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\Big) \tag{35}$$

This is the same advantage as in Case 1 and hence Therorem 1 follows.

B Proof of Theorem 2:

The security difference between Protocol 1 and 2 is that the latter provides full WFS, i.e. in addition to the adversarial capabilities considered in the proof of Theorem 1, we now allow the adversary to corrupt both parties to the test session. It is natural then to consider the proof of Theorem 2 in two parts: the first part where the adversary does not corrupt both parties to the test session, and the second part where it does. Then, the first part is essentially identical to the proof of Theorem 1. The only difference is that in the analysis of Game 3 (in both Case 1 and 2) $\mathcal{A}$ needs to simulate the extra Diffie-Hellman values, which $\mathcal{A}$ can easily do for all sessions, including the test session. (Note that $\mathcal{A}$ will always choose at least one of Y_A or Y_B, hence it can always compute K''_{AB}).

We deal now with the second part of the proof, where the adversary corrupts the two partners to the test session. Note however that the adversary is restricted to being passive during the protocol run corresponding to the test session – a consequence of only being able to achieve weak forward secrecy in one round. As we will see below this allows us to inject a challenge Decisional Diffie-Hellman triplet into the test session.

The second part of the proof allows any party to be corrupted. It considers the following four games with $\mathcal{B}$.

Game 0. This game is the same as a real interaction with the protocol. A random bit b is chosen, and when $b = 0$, the real key is returned in answer to the test session query, otherwise a random key from $\mathbb{U}_2$ is returned.

Game 1. This game is the same as the previous one, except that before the adversary begins, random values $j, j^* \stackrel{\$}{\leftarrow} \{1, 2, \ldots, n_{\text{orac}}\}$ are chosen. Let T and T^* be the owners of the j^{th} and $j^{*\text{th}}$ sessions respectively. If the j^{th} session at T is not the test session or if the output of the $j^{*\text{th}}$ session at T^* is not used as input to the test session, then the protocol halts and $\mathcal{B}$ fails and outputs a random bit. Furthermore, the session key of the test session is calculated as usual except that $\mathsf{Exct}(h)$ for $h \in_R \langle f \rangle$ replaces K''_{TT^*}. The test session's matching session has its key set to the same value (i.e. a random test session key is always returned, no matter what the value of b).

Game 2. This game is the same as the previous one except that $\mathsf{Exct}(h)$ is replaced with $K'' \in_R \mathbb{U}_1$

Game 3. This game is the same as the previous one, except that whenever the value $\mathsf{Expd}_{K''}(s')$ for any s' would be used in generating keys, a random value from $\mathbb{U}_2$ is used instead (the same random value is used for the same value of s'; a different random value is chosen for different s').

B.1 Analysis Game 1

We now construct adversary $\mathcal{A}$ against the DDH problem, using $\mathcal{B}$. $\mathcal{A}$ is constructed such that provided $\mathcal{A}$ does not have to abort the protocol as specified in Game 1 then if $\mathcal{A}$'s input is from $\mathcal{DH}_F$, the view of $\mathcal{B}$ is the same as in Game 0, but if $\mathcal{A}$'s input is from $\mathcal{R}_F$, the view of $\mathcal{B}$ is the same as in Game 1. Then, by Assumption 4, we can claim these games are indistinguishable.

To begin, $\mathcal{A}$ generates all the protocol parameters and passes the public parameters to $\mathcal{B}$.

Let (f^a, f^b, h) be $\mathcal{A}$'s challenge DDH inputs (with h either f^c or f^{ab}). When the j^{th} and the $j^{*\text{th}}$ sessions are activated, $\mathcal{A}$ uses its inputs f^a and f^b instead of the values Y_T and Y_{T^*} when it generates the outputs of these sessions. Apart from this change, all session inputs and outputs are generated according to the protocol specification.

When the test session query is made, $\mathcal{A}$ uses $\mathsf{Exct}(h)$ in place of K''_{TT^*} when calculating the real test session key.

$\mathcal{A}$ is able to answer all other queries correctly since it knows all of the system parameters and all of the session states. $\mathcal{A}$ outputs 1 if $\mathcal{B}$ is correct, 0 otherwise.

The probability that $\mathcal{A}$ will not have to abort the protocol as described in Game 1 is $\frac{1}{n^2_{\text{orac}}}$. Hence, by the game hopping technique from Dent (2006) and using a similar logic to that shown in (26) to (29) we have that:

$$\tau_0 \leq 2n^2_{\text{orac}}\mathbf{Adv}^{\text{ddh}}_{F,\mathcal{D}}(k) + \tau_1 \tag{36}$$

B.2 Analysis Game 2

The analysis of Game 2 is the same as that of Game 4 of Case 1. Thus,

$$\tau_1 \leq 2\epsilon + \tau_2 \tag{37}$$

B.3 Analysis Game 3

The analysis of Game 3 is the same as that of Game 5 of Case 1 and 2. Thus,

$$\tau_2 \leq 2\mathbf{Adv}^{\text{p-rand}}_{\mathcal{F},\mathcal{C}}(k) + \tau_3 \tag{38}$$

Since the test session key is masked with a random value, and that value is independent of all other sessions, $\mathcal{B}$ has no advantage in Game 3 ($\tau_3 = 0$). The value is independent because only the test session has the test session's session id.

B.4 Combining Results:

Let E be the event that the test session has a matching session by the time $\mathcal{B}$ finishes, and let σ be the event that $\mathcal{B}$ guesses b correctly in Protocol 2. Then we have:

$$\begin{aligned}\mathbf{Adv}^{\text{sk}}_{\mathcal{B}}(k) &= |2\Pr[\sigma] - 1| \\ \Pr[\sigma] &= \Pr[\sigma|E]\Pr[E] + \Pr[\sigma|\neg E]\Pr[\neg E] \\ &= \Pr[\sigma|\neg E] + \Pr[E](\Pr[\sigma|E] - \Pr[\sigma|\neg E]).\end{aligned}$$

Therefore,

$$\begin{aligned}\min(\Pr[\sigma|\neg E], \Pr[\sigma|E]) &\leq \Pr[\sigma] \\ &\leq \max(\Pr[\sigma|\neg E], \Pr[\sigma|E])\end{aligned}$$

and

$$\mathbf{Adv}^{\text{sk}}_{\mathcal{B}}(k) \leq \max\left(\mathbf{Adv}^{\text{sk}}_{\mathcal{B}}(k)\,|E,\ \mathbf{Adv}^{\text{sk}}_{\mathcal{B}}(k)\,|\neg E\right)$$

and so we can combine (30), and (35)-(38) to find:

$$\begin{aligned}\mathbf{Adv}^{\text{sk}}_{\mathcal{B}}(k) \leq \\ \max\Big(&2n^2_{\text{orac}}\mathbf{Adv}^{\text{ddh}}_{F,\mathcal{D}}(k) + 2\epsilon + 2\mathbf{Adv}^{\text{p-rand}}_{\mathcal{F},\mathcal{C}}(k), \\ &\frac{n^2_{\text{orac}}}{b} + 2n_{\text{orac}}\Big(\mathbf{Adv}^{\text{CL-KEM-CCA}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \\ &\mathbf{Adv}^{\text{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\Big)\Big).\end{aligned}$$